NAG 1-775

IN-61-CR

252408

29P

# Collision Avoidance Algorithms for a Telerobotic Environment

William L. Bynum
College of William and Mary

Nancy E. Sliwa
NASA Langley Research Center

## ABSTRACT

The purpose of this paper is to enumerate a number of collision avoidance algorithms for robotic manipulators and consider them in the context of a teleoperator environment, such as the one found in the Intelligent Systems Laboratory at NASA/Langley Research Center.

Collision detection and avoidance algorithms can be separated into three broad categories: *environment modelling*, *external control function*, and *free space calculation*. This paper discusses two selected environment modelling systems, two external control function systems, and three free space calculation systems in the context of the telerobotic environment. An extensive bibliography of collision avoidance work is included. An arrangement of the bibliography by topic is included as an appendix.

## INTRODUCTION

The purpose of this paper is to enumerate a number of collision avoidance algorithms for robotic manipulators and consider them in the context of a teleoperator environment, such as the one found in the Intelligent Systems Laboratory at NASA/Langley Research Center.

A number of authors have proposed collision avoidance algorithms, and the terminology that they use varies slightly from one another. The reader might find the following brief discussion of terminology helpful to clarify the way terms are used in this paper.

*Collision detection* is the detection of a collision between a robotic manipulator and an object in its environment. To be useful in predicting collisions with objects in the environment, collision detection is typically based on a future position of the manipulator.

*Collision prediction* is the prediction of an impending collision between a manipulator and an object in its environment. It is one level higher than collision detection the sense that collision prediction is just collision detection applied to the future positions of the manipulator and the objects in its environment.

(NASA-TM-101270) COLLISION AVOIDANCE ALGORITHMS FOR A TELEROBOTIC ENVIRONMENT (NASA) 29 p

N90-70381

Unclas
00/61 0252408

*Collision avoidance* is the act of avoiding collisions between a manipulator and the objects in its environment. *Obstacle avoidance* is a synonym, because *obstacle* is typically used to refer to an object (usually stationary) in the environment with which the manipulator could collide. Usually, a collision detection or prediction scheme is used to implement the collision avoidance capability.

*Path planning* refers to the action of planning a collision-free path for the robot from a starting point to a goal. Path planning presumes some sort of collision avoidance capability, and typically does not include any time parameterization of the path (see *trajectory planning* below). Lozano-Pérez [31] coined the phrase *Findpath Problem* to characterize the problem of planning a path for a robot through a collection of obstacles. Usually, it is assumed that the obstacles will be stationary, particularly in the earlier research. Much research, particularly the systems that calculate free space explicitly, were developed with the goal of automatic path generation. There has been little research on path planning or collision avoidance from the point of view of teleoperated robotic manipulators.

*Trajectory planning* consists of creating a detailed specification of the motion of a manipulator that will cause it to proceed from an initial position to a goal position and usually involves some specification of the time parameters of the path. Some authors use *trajectory planning* as a synonym for *path planning*, but others give the term a more technical meaning as a specification of a succession of via points (or joint angles, velocities, or torques) to achieve a desired path by a manipulator. In this paper, trajectory planning will be taken to mean a specification of a path through the use of a time parameter.

## REQUIREMENTS OF A TELEOPERATOR ENVIRONMENT

Since the purpose of this paper is to analyze and discuss collision avoidance schemes from the perspective of the use of the algorithm with a robotic manipulator in a teleoperator environment (or *telerobotic environment*, for brevity), a brief discussion of the requirements of such an environment will be helpful.

The telerobotic environment is a *dynamic* environment, typically containing multiple moving objects (robotic manipulators and possibly other mobile objects), and stationary objects. To provide useful information to the operator, the collision avoidance system should monitor manipulator state (joints, velocities) under control of the operator, as well as the location of the other objects in the environment, then synthesize and present this information to the operator in real time. Moreover, this facility should be provided to the operator without overly burdening the executive controller of the manipulator system, which in the case of the ISRL is the VAX 750. The collision avoidance system should not complicate the operator interface, but instead should be a helpful adjunct for the operator.

The path specified by a task, either by the operator or an executive path planner, should not be required to be fixed, but should be alterable in response to changes in the environment. All links and joints of all manipulators should be considered, not just the position of the end effector. Some telerobotic applications, particularly those located in space, require that jerk, acceleration, and deceleration be minimized. Since a telerobotic manipulator may need to handle objects in its environment, the collision avoidance system used must be able to deal easily with the "benign" collisions that are required in grasping or mating objects, inserting pegs, and other specialized operations.

Finally, the collision avoidance system must be sufficiently flexible to accommodate changes in the environment, such as relocation of the manipulators or other objects, without requiring modification of the system.

## Types of collision detection and collision avoidance algorithms

Collision detection and avoidance algorithms can be separated into three broad categories:

- environment modelling
- external control function
- free space calculation

*Environment modelling* systems usually use some sort of geometric approximation, such as cylinders, convex polygons, or spherelists, to model the objects in the environment. Velocities of the mobile objects in the environment are used to use the calculated future positions of the objects in the environment to predict collisions that might occur. These systems do well in an uncluttered environment because of their computational and conceptual simplicity. *Every* collision avoidance algorithm must have the ability to model the geometry of the environment, so in this sense, environmental modelling systems are the most fundamental of all of the collision avoidance systems.

*External control function* systems rely on an external function, such as a potential field of attractive and repellent forces, a penalty function, or a higher-level coordinator, to control the position of the manipulator in the environment. In these systems, collision avoidance is moved from the level of manipulator control to an external agent which ensures that collisions do not occur. For a higher-level coordinator, its level in the hierarchy ensures that the coordinator has the knowledge of the environment to ensure that collisions do not occur. In the case of a potential field or penalty function, collision avoidance becomes a low-level, autonomic reflex. This makes it possible to specify a path in terms of general goals, leaving it to the autonomic avoidance behavior to ensure that collisions do not occur.

*Free space calculation* systems calculate explicitly the areas (or volumes) in the environment in which collisions with obstacles is impossible — the "free space" in the environment. A collision-free path is then constructed through the free space by various methods. The principal disadvantage of this type of system is the computational expense of the explicit calculation of free space. This type of system has generated a large amount of research, because a path planning method based on free space calculation is *guaranteed* to find a collision-free path from an initial point to a goal point, if one exists. Furthermore, the set of all such paths can be analyzed to select a path that is minimal with respect to some constraint, such as path length or joint torques.

## DISCUSSION OF SPECIFIC COLLISION AVOIDANCE SYSTEMS

Each of the following three sections are devoted to a discussion of representative examples of these three different types of collision avoidance algorithms. The examples will be organized by the last names of their author or authors. The discussion of each example will include

- a description of the system
- testing that the authors performed on their system
- a general evaluation of the system's strengths and weaknesses
- an evaluation of the suitability of the system for use in a telerobotic environment

The Appendix contains the entries of the Bibliography, grouped according to these three categories of collision avoidance systems. There are a number of entries in the bibliography that do not fall into one of the above three categories. The phrase "collision avoidance" includes path planning problems of a general nature, such as the "piano movers" problem. This problem involves describing a general algorithm to move an irregular shape through a narrow passageway. A number of theoretical results have been obtained on this problem and others, and are grouped as a separate category in the listing in the Appendix.

## ENVIRONMENT MODELLING SYSTEMS

There are two recent environmental modelling systems, the motion simulation paper of Uchiki, Ohashi, and Tokoro, and the spherelist modelling system developed by Wallace, Sliwa, and Bynum.

### Uchiki, Ohashi, and Tokoro

Description of the system. The system was developed for a computer animation system to determine when two points occupy the same point in space [57]. The authors developed the system to prevent graphic objects from appearing to jump over or pass through each other.

The authors represent points in three-dimensional space as a block of points in a Picture Element Array (PEARY). In their terminology, the word "object" is taken to be synonymous with "named point", so that at first glance the system appears more grand than it actually is. The system has three major components:

(1) Object Name Table
(2) Access Controller
(3) Message Manager

The Object Name Table is simply an array of object names. The index of an object's name in the Object Name Table is used as the object's ID. The Access Controller transfers objects into and out of the PEARY. When an object is written into PEARY and another object already occupies the point in space for which the incoming object is targeted, the Access Controller discovers a collision and passes this information on to the Message Manager. The Message Manager maintains a Collision Information Table. This table is arranged by the coordinates of each point in space at which a collision has occurred, along with the ID's of the objects occupying that point. There is room for two ID's in the table (this is probably the most frequently occurring case). If more than two objects occupy the same point in space, the overflow is placed in a linked list off of the table entry.

Testing that the authors performed on their system. It is unclear from the paper that the authors have actually used the system. The authors claim to have evaluated the system using a "simulation" written in C, running on UNIX 4.1BSD hosted on a VAX 11/750. Their simulation uses 2-D cells which can be assigned 256 different "priorities", which they describe as a 2.5D system. The number of collision detectable objects at each priority level is limited to eight. Their simulation shows that their system is effective. There are no pictures or diagrams in the paper that appear to be taken from the system itself. The illustrations in the paper appear to be produced by hand.

Evaluation of the system's strengths and weaknesses. The principal strength of the system is its apparent speed. The authors indicate that the collision detection cycle time is approximately equal to the time required to write to their memory.

The system has two serious weaknesses. First, it deals only with *points*. There is no discussion in the paper about how to use the system to deal with more complicated geometric shapes, such as spheres, cylinders, and convex polyhedra. There are several illustrations in the paper that seem to intimate that the system could be used in this way, but this point is not supported by the program text. Second, the system has only been *simulated* in 2D. The discussion in the text is in a three-dimensional context, and the extension of the system to three dimensions does not seem to pose insurmountable difficulties. Nevertheless, the authors have not yet extended the simulation to three dimensions, and it is not clear that the authors have actually used the system in two dimensions.

Suitability of the system for use in a telerobotic environment. The system is not suitable for the telerobotic environment. It is limited to points only. The representation of more complex geometric shapes would be prohibitively expensive, both in terms of space and time. The additional space required for more complex objects would also require additional time for processing.

No demonstration of the capability of the system to deal with three-dimensional problems has been conducted.

Description of the system. This is a collision detection system based on the spherelist object representation of the PUMA environment that was developed by Richard Wallace and extended by Nancy Sliwa. The system was designed to detect collisions between a PUMA in the ISRL and other objects in the environment. The basic spherelist structure was used in a demonstration of two cooperating PUMA manipulators by Wallace in 1983. Each link of each PUMA was approximated by a spherelist. Figure 1 below shows the approximation of a PUMA manipulator by several spherelists.

In 1987, James Reich, an aeronautical engineering co-op student from M.I.T., conducted another proof-of concept demonstration of the collision detection system. Reich's program was written in C, ran on one of the micro-VAX's and communicated with the VAX 750 hosting the REALTIME program via DECNET. Reich's work was useful in demonstrating that a collision detection system based on the spherelist method of geometric modelling could return results sufficiently quickly to be useful. The limitations of Reich's program prevent its further development. The positions of the PUMA's in the ISRL were hard-coded in the program. Furthermore, it was assumed throughout the program that this placement of the PUMA's guaranteed that only the end effectors could collide. The program did not allow the introduction of other mobile objects into the environment. Although the program allows stationary objects other than the PUMA's in the environment, this feature was apparently not tested.

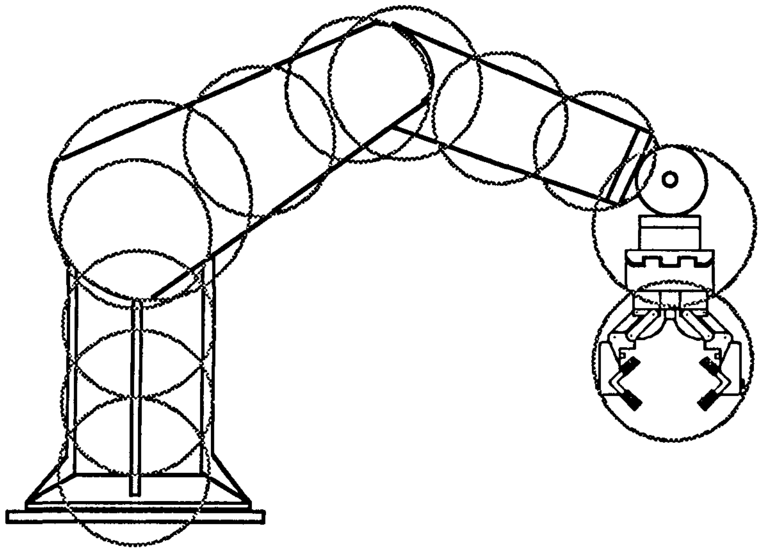

Figure 1. Representation of a PUMA manipulator as a spherelist

The construction of a more general spherelist collision detection system was begun in 1987 – 1988 by Bill Bynum, working with Nancy Sliwa. The system is written in VAX COM-

MON LISP. Porting of the program to both the Symbolics and the VAX AI workstation for use with the REALTIME system is planned for 1988 – 1989. The system allows the placement of any number of mobile and stationary objects in the environment. The system is highly modular. Each object in the environment is represented by a code module that is responsible for providing the system with the functions necessary to provide information to the system about the geometry of the object and its laws of motion.

Every object in the system is represented as a spherelist. A PUMA manipulator is represented as a collection of objects (its links 1 through 5) that move in concert. Each object (or collection of objects, as with a PUMA) is expected to provide the system with two classes of functions: an *initialization* function that establishes the initial position of the object, and an *update* function that predicts the position of the object based on the basis of the object's previous position and current velocity.

The system maintains three global data structures, used for collision detection:

- Collision Array.

  An *n* by *n* lower-triangular array used to record the last known distance apart for each pair of objects in the environment (here, *n* is the maximum number of objects expected in the environment – currently thirty). The lower-triangularity of the array is used to ensure that collision checking between each pair of objects occurs no more than once; that is, the array is used to determine which pairs of objects can collide (no collision checking between two stationary objects is required, for example), and that collision checking is performed only once for those pairs of objects that *can* collide.
- Mutual Distance Travelled Array

  An *n* by *n* lower-triangular array used to record the cumulative distance travelled for each pair of objects (at least one of which is mobile) since the last collision check. A full collision check between the spherelists of the two objects is only triggered when the entry in this array exceeds the last known distance between them stored in the Collision Array. This action has been added to reduce the amount of collision checking required. When a collision check occurs for two objects, the corresponding entry of this array is set to zero. A collision check that reveals an impending collision stimulates a warning message.
- Distance Travelled Array

  An *n* by *1* vector composed of the distance that each object has travelled in the last time interval. The array is used to calculate the Mutual Distance Travelled Array.

The system maintains a graphics display on the VS11 graphics terminal showing the side and top view of the environment. This display is useful for determining the current positions and movement of the objects in the environment.

Testing that the authors performed on their system. The system has been subjected to only a limited amount of testing. The graphical display is used to monitor the three-dimensional position of all objects in the environment and to give a separate indication of whether a collision should have occurred. The system uses the current locations of the PUMA manipulators, which makes collision between them relatively unlikely. To test the collision detection capabilities of the system, two different types of objects were introduced: a *random* object that moves a random distance from its current position at each time interval, and a *magnet* object that can be locked onto another object so that it will halve its distance from the chosen object at each time interval. More testing of the system needs to be done. More realistic testing with the REALTIME system can occur after the system has been ported to the Symbolics and VAX AI workstation this summer.

Evaluation of the system's strengths and weaknesses. The strengths of the system are its generality, its highly modular structure, and its conceptual simplicity. The generality of the system stems from the freedom of constraints that the system places on the initialization and update functions for a given object. The update function for an object can use whatever parameters and rules of motion that are most appropriate for the object. The initial position of an object is not constrained in any way, and the object can declare itself to be "mobile" or "fixed". The modularity of the system follows from the fact that each code module is independent from the modules of the other objects. The conceptual simplicity of the system follows from the geometric simplicity of a sphere.

The principal weakness of the system is the fact that it has not been fully tested in a working laboratory environment. Some authors have suggested that collision detection through geometric modelling will be unable to handle a cluttered environment successfully. Without thorough testing, it is unclear whether other systems with similar capabilities will perform significantly better in a cluttered environment.

Suitability of the system for use in a telerobotic environment. This system has considerable interest for application in the ISRL telerobotic environment, both because of its familiarity and the preliminary proof-of-concept demonstrations that have already occurred. Only time and further testing will determine whether other systems of comparable capability are superior.

## EXTERNAL CONTROL FUNCTION SYSTEMS

The most widely known external control function system is the system of Khatib using an artificial potential field [24,25,26,27,28]. Freund and Hoyer have described an external control function system for multiple robotic manipulators [15,16,17,18].

### Khatib

Description of the system. Khatib originated the idea of using an artificial potential field to control a robotic manipulator. His system grew out of his work on end effector motion control amid obstacles on the Montpellier manipulator in 1978. He has published several papers on the system. Khatib's system moves collision avoidance from a high planning level down to low-level control as a sort of "autonomic" response. Khatib holds that the planning level is slow, whereas the lower level can respond much more quickly.

Khatib uses what he terms the "operational space formulation" of controlling a manipulator, which is used synonymously with his potential field control system. The operational space formulation is analyzed in [24,25,26,27,28]. The analysis is similar in all of these references. In [26], the most accessible of these references, the operation space formulation is analyzed for the special case of a manipulator end effector as a single unit mass with a single obstacle.

The position of the end effector is described by a vector **x** of $m_0$ coordinates. The kinetic energy of the articulated mechanism is characterized by a quadratic form, *T*, involving **x** and **x'** . A Lagrangian equation is used to obtain the following differential equation characterizing the motion of the end effector in the potential field:

$$\Lambda(\mathbf{x})\,\mathbf{x}'' + \mu(\mathbf{x},\mathbf{x}') + \mathbf{p}(\mathbf{x}) = \mathbf{F}$$

where $\Lambda$ is the symmetric matrix of the quadratic form *T*, $\mu$ is a function representing the centrifugal and Coriolis forces on the manipulator, **p** is a function representing gravity forces, and **F** is the potential force vector.

An "artificial" potential field function **F** is developed to be plugged into the above equation so that the equation can be solved to determine the position of the manipulator. The goal position of the end effector is assumed to be an attractive pole in the potential field and the obstacle is assumed to be a repulsive surface in the field, and the total force acting on the end effector is the sum of the two forces. The attractive potential due to the goal is modelled as a PD (proportional-derivative) servo with a friction (or *dissipative*) term. The repulsive potential due to the obstacle is a non-negative continuous, differentiable function whose value tends to infinity as the end effector approaches the obstacle's surface, and is limited to a given region surrounding the obstacle ("to avoid undesirable perturbing forces beyond the obstacle's vicinity"). The function used expresses the force in terms of the square of the difference of the reciprocals of the shortest distance to the obstacle $\rho$ and the limit of influence of the obstacle $\alpha$:

$$\begin{array}{c} K \cdot (1/\rho - 1/\alpha)^2, \text{ for } \rho \le \alpha, \\ 0\,, \text{ for } \rho > \alpha. \end{array}$$

Obstacles are modelled by means of combinations of primitive shapes: rectangular parallelipiped, cylinder, cone, and "n-ellipsoid" (equation like an ellipse but with an exponent of $2n$ instead of $2$). Collision avoidance of links is taken care of by approximating links with straight lines and "continuously controlling the link's closest point to the obstacle". Joint constraints can also be implemented by a repulsive potential, like an obstacle. Formulas for the minimum distances from a link to the primitive shapes of a parallelipiped, cylinder, and cone are given in appendices of [26].

Testing that the author performed on his system. In [26], Khatib alludes to a videotaped demonstration "of real-time collision avoidance with links and moving obstacles ... performed using a PUMA 560 and a Machine Intelligence Corporation module." Although the authors of the present paper have not been able to view this demonstration, we understand from others who have seen it that the obstacle avoidance displayed there is essentially two-dimensional. Khatib claims to achieve a servo control rate of 225 Hz and a coefficient evaluation rate of 100 Hz.

Evaluation of the system's strengths and weaknesses. The principal strength of this system is its conceptual elegance. It is very attractive to be able to move collision avoidance from the high, planning level down to the control level. The higher, slower planning level is unburdened of the lower-level details of preventing collisions between the manipulator and obstacles in the environment.

The principal weakness of the system is best phrased by Khatib himself in [26]:

"complexity of tasks that can be implemented with this approach is limited. In a cluttered environment, local minima can occur in the resultant potential field. This can lead to stable positioning of the robot before reaching its goal."

The computation involved seems complex and might consume appreciable amounts of time. It is unclear how the system would perform in a cluttered environment. The details of the analysis of a real manipulator with moving objects in a three-dimensional environment is not given. Almost no implementation details are given in any of Khatib's papers. It is unclear whether the repulsive functions used to model the forces from obstacles are satisfactory in all situations.

Suitability of the system for use in a telerobotic environment. The conceptual elegance of using an artificial potential field to handle collision avoidance makes this system very attractive. By moving the collision avoidance down to a lower, autonomic level, not only is the higher level of the controlling program unburdened, but so also is the teleoperator, if one is involved. The lack of specific implementation details in Khatib's papers make it difficult to evaluate the algorithm accurately. It would be very interesting to implement this algorithm in the ISRL and compare it with other methods.

### Freund and Hoyer

Description of the system. In a series of papers [15,16,17,18], Freund and Hoyer have described a collision avoidance system for multiple robotic manipulators that, like the system of Khatib, has an external control function. The control function for their system is called the "Hierarchical Coordinator". The action of the Hierarchical Coordinator are determined by a complicated system of nonlinear vector feedback equations. These equations, as well as the concept of the Hierarchical Coordinator, are most easily understood with the help of diagram shown in Figure 2, which shows the system of equations graphically. This diagram was adapted from Figure 1 of [18].

The system is assumed to have *r* robotic manipulators. The position of the *k*-th robot is described by the position vector $x_k(t)$. The *k*-th robot obeys a vector differential equation involving $x_k$.and its first derivative, and the vector functions, $A_k(x_k)$, $B_k(x_k)$, and $u_k(t)$:

$$x_k'(t) = A_k(x_k) + B_k(x_k)u_k(t).$$

This equation appears two times, once for each robotic manipulator, in Figure 2. In the equation, $A_k(x_k)$ is a matrix characterizing the dynamic behavior of the robot, $B_k(x_k)$ is the input matrix, and $u_k(t)$ is the input vector.

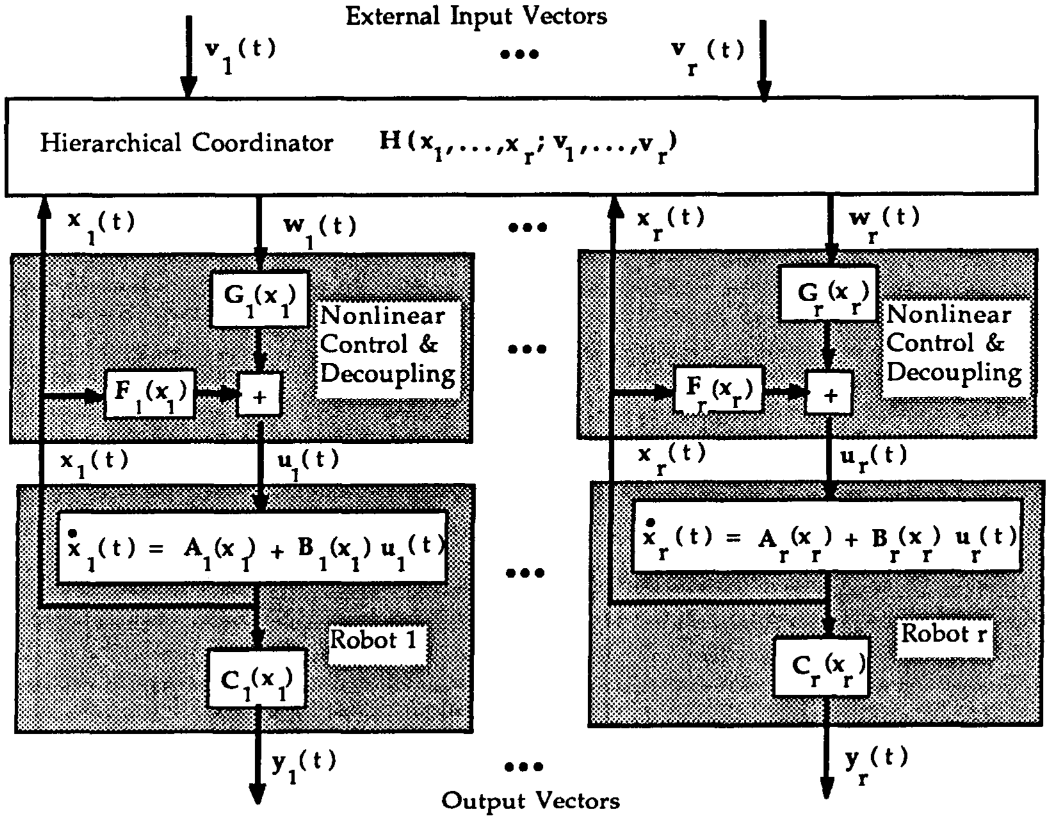
External Input Vectors
$v_1(t)$
$v_r(t)$
Hierarchical Coordinator $H(x_1,\ldots,x_r; v_1,\ldots,v_r)$
$x_1(t)$
$w_1(t)$
$x_r(t)$
$w_r(t)$
$G_1(x_1)$
$F_1(x_1)$
+
Nonlinear Control & Decoupling
$G_r(x_r)$
$F_r(x_r)$
Nonlinear Control & Decoupling
$u_1(t)$
$u_r(t)$
$\dot{x}_1(t) = A_1(x_1) + B_1(x_1)u_1(t)$
$\dot{x}_r(t) = A_r(x_r) + B_r(x_r)u_r(t)$
Robot 1
Robot r
$C_1(x_1)$
$C_r(x_r)$
$y_1(t)$
$y_r(t)$
Output Vectors

Figure 2. Hierarchical Coordinator feedback diagram

The input vector $u_k(t)$ is decomposed further into nonlinear and control terms by the equation:

$$u_k(t) = F_k(x_k)+ G_k(x_k)w_k(t).$$

The terms $\mathbf{F}_k$ and $\mathbf{G}_k$ are chosen to decouple the inputs and outputs to the differential equation above and place the poles of the equation as desired. The $\mathbf{w}_k(t)$ term is the "reference input" term supplied as an output of the Hierarchical Coordinator.

The exact behavior Hierarchical Coordinator function is derived using what Freund and Hoyer call the "collision trajectories" of the robotic manipulators; namely, the geometry of the manipulators and their intended trajectories. For a given manipulator k, this information is contained in the external input vector $\mathbf{v}_k(t)$ and the position vectors $\mathbf{x}_k(t)$. The authors claim that "vertical movement is not considered in the avoidance strategies, but it can be involved without principal difficulties". This means that the Hierarchical Coordinator, as analyzed in [18], deals only with *two*-dimensional trajectories.

Testing that the authors performed on their system. The system has never been tested in an actual robotic environment. The behavior of the system has been *simulated* using the simulation language SIMWAIT at the Institut für Roboterforschung in West Germany. Simulations were performed for four cases:

1. moving robot arms, stationary manipulators, and stationary, constant-sized obstacles.
2. stationary robot arms, moving manipulators, and moving, constant-sized obstacles.
3. moving robot arms, moving manipulators and moving, variable-sized obstacles.
4. moving robot arms, moving manipulators & moving, constant-sized obstacles.

Since the algorithm was only simulated for these four cases, the computation times required by the algorithm for the four cases were *estimates* only. The results were presented in terms of "time units". No explanation of the relationship of these time units to real time was given, although the authors remark that for the TI 9995, Case 1 is estimated to require 9 milliseconds, and Cases 2, 3, and 4 are estimated to require 10 milliseconds.

Evaluation of the system's strengths and weaknesses. The principal strength of this paper is that it is a recent solution to the Findpath problem for multiple manipulators. The authors claim that it is fast.

This system has a number of weaknesses. It has as all of the technical complication of Khatib's work (perhaps a great deal more) with none of the conceptual elegance. Because of the exceptional technical complexity of the algorithm and the fact that the system has only been simulated and never demonstrated in hardware, it seems probably that the algorithm is computationally intensive. In spite of the author's estimates, it is unclear how fast the algorithm will turn out to be in practice. The technique of developing the Hierarchical Coordinator for an arbitrary system of manipulators would be difficult. It is unclear how difficult it would be to modify the Hierarchical Coordinator when changes in the environment were required. The question of how well the simulation models an actual system of manipulators is not dealt with.

Suitability of the system for use in a telerobotic environment. It is not clear how a teleoperator would share control of the system with the Hierarchical Coordinator. The teleoperator would probably tend to get in the way of the Hierarchical Coordinator. The question of how to modify the design of the Hierarchical Coordinator to include the effects of teleoperator control might be difficult. The technical complexity of the algorithm would make actual implementation (as opposed to simulation) difficult.

## FREE SPACE CALCULATION SYSTEMS

*Free space* is the space in a robotic environment not occupied by either the robot or its obstacles. Free space calculation systems are all based on the presumption that path planning is a high-level, infrequently performed activity, because the calculation of free space is a time-consuming, computationally expensive activity. This fact usually leads to the assumption that obstacles in the environment are stationary, since free space must be recalculated when any obstacle moves.

There extensive amount of research devoted to this type of collision avoidance. The free space calculation systems are ill-suited to a teleoperator environment, however, because they are all based on the premise that the path planning will be done automatically and infrequently and that changes in the environment are rare. There is no allowance for a teleoperator in the path planning. These facts, along with the high cost of the explicit calculation of free space, support the conclusion that free space calculation methods cannot be used without substantial modification in a telerobotic environment like the ISRL. These papers are discussed here because of their historical significance and their general influence on collision avoidance.

The paper of Lozano-Pérez and Wesley [35] is one of the first papers on free space calculation. It has had a profound influence on all subsequent free space calculation systems. The papers of Brooks [6,7] involve *freeways*, an improvement on the free space calculation method of Lozano-Pérez and Wesley.

The paper of Gouzènes [19] is interesting because it is a recent study of free space calculation.

### Lozano-Pérez and Wesley

Description of the system. This is the best known and most widely referenced collision avoidance scheme.

The algorithm is based on the assumption that all obstacles are polygonal in shape. The algorithm has the following steps:

1. Represent the movable part of the manipulator (usually the end-effector) by a sphere.
2. Shrink the end-effector to a point and grow all obstacles by the radius of the sphere in step 1.
3. Construct an undirected graph with a vertex set consisting of the starting point, S, the goal point, G, and the set of vertices of all (grown) obstacles in the environment and an edge set consisting of all straight-line segments from one member of the vertex set to another that do not overlap an obstacle. This graph is called the *Visibility Graph.*
4. The shortest collision-free path from S to G in the plane is then the shortest path in the Visibility Graph from the vertex corresponding to S to the vertex corresponding to G.

This algorithm is guaranteed to find a shortest path from the start point to the goal point, if any such path exists. Figure 3 below illustrates the algorithm.

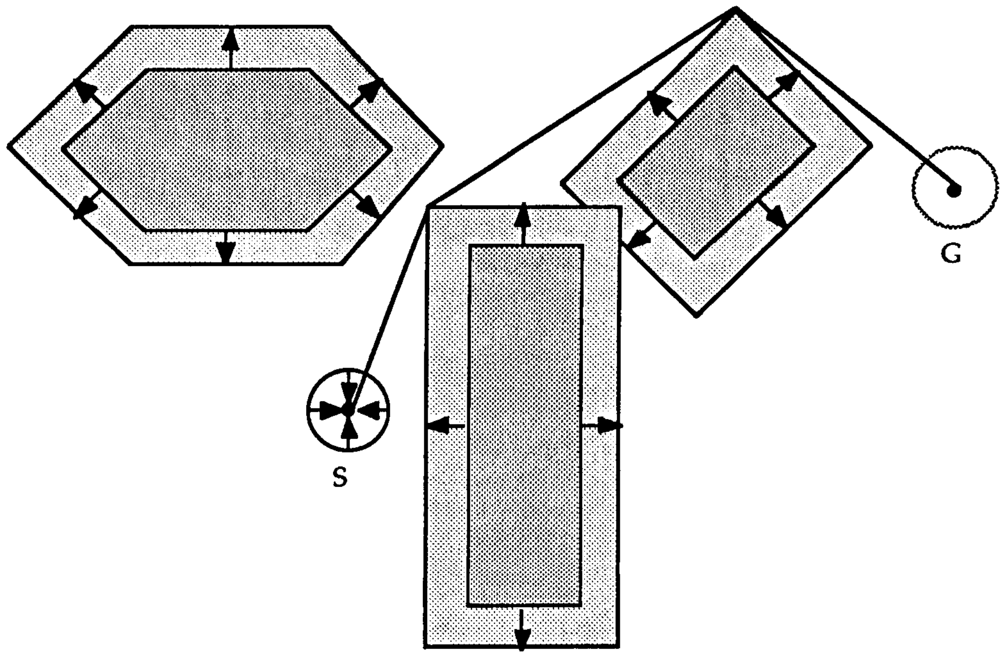

Figure 3. Visibility Graph solution to the Findpath problem.

In subsequent papers [32,33], Lozano-Pérez supplied a theoretical framework for considering the Findpath problem that he calls *Configuration Space.* Lozano-Pérez points out that a *configuration* of an object can be specified by a six-dimensional vector composed of its three dimensional cartesian position, along with the object's Euler angles. Then the *Configu-*

*ration Space* for an object is just the collection of all of its possible configurations. In spite of the specific, technical meaning that Lozano-Pérez assigns to *Configuration Space,* this term has come to be associated with the procedure described above of shrinking the movable part of the robot to a point and enlarging the obstacles.

John Canny, working under Lozano-Pérez, received the ACM Doctoral Dissertation Award in 1987 for his work on the computational complexity of the Lozano-Pérez configuration space approach to robot motion planning [10]. Canny develops methods that reduce the computational burden of the configuration space approach to collision avoidance. In spite of these methods, Canny shows that the problem of motion planning for a moving robot with moving obstacles is NP-hard. Furthermore, the configuration space method of collision avoidance still has the problem that the configuration space characterization of the environment changes when the end effector grasps an object and free space must be re-calculated.

Testing that the authors performed on their system. The authors mention briefly that they have implemented the algorithm in PL/I on an IBM 370/168 and have used it to plan collision-free "trajectories" for a seven degree-of-freedom computer controlled manipulator. The word "trajectory" as it is used by these authors should probably be replaced by "path", since "trajectory" is usually taken to connote a parameterization of the path by time, which is not a part of this algorithm.

Others have implemented the algorithm, too. For instance, Don Krause, as his summer project at the ISRL, simulated this algorithm on the VAX [29].

Evaluation of the system's strengths and weaknesses. The strengths of the system are its conceptual, theoretical, and computational simplicity. Furthermore, like almost all of the other free space calculation systems, this algorithm is guaranteed to find a collision-free path from a starting point to a goal point, if such a path exists. Once the free space has been calculated, then it is possible to select a path from all of the possible paths that is minimal in some respect, such as total distance travelled, joint torques, and so forth.

Unfortunately, the simplicity of this algorithm is also its most serious problem. The only mobile object in the environment is assumed to be the end-effector, and moreover, its shape is assumed to be spherical. Lozano-Pérez and Wesley discuss the use of other shapes in an appendix to their paper, but the use of more complicated shapes increases considerably the computational complexity of the algorithm. The basic algorithm is only two-dimensional. Others, such as Brooks (see below) have extended the algorithm to three-dimensions.

The path produced by the algorithm is placed *as close as possible* to the obstacles that the end-effector passes on its way to the goal. Other algorithms, such as Brooks (see below) have been developed that center the path between the obstacles in free space.

Like all of the other free space calculation systems, this algorithm has trouble if the obstacles move, since this necessitates recalculation of the free space. Inclusion of the other links of the manipulator increases substantially the computational burden.

Suitability of the system for use in a telerobotic environment. The system was never intended for use in a teleoperator environment. The teleoperator is not needed. Path planning is done automatically from a map of the environment. Intervention of a teleoperator is neither needed nor allowed for. Modification of the path according to constraints supplied by a teleoperator at run time (or path traversal time) is not allowed.

Brooks

Description of the system. This algorithm is an improvement of the Lozano-Pérez and Wesley algorithm, in that it constructs a path on which the robot stays as far as possible from the obstacles in the environment. Brooks' algorithm, as stated in [7], is a direct derivative of the Lozano-Pérez and Wesley algorithm, because it also assumes polygonal obstacles, it also shrinks the robot to a point, and it is also two-dimensional. In a slightly later MIT AI memo [6], Brooks describes a way of extending his freeways to three dimensional obstacles and a four-link manipulator.

Brooks constructs a collision-free path in free space using *freeways*. A freeway is a "generalized cone" in free space formed by opposing faces of two separate obstacles. The *spine* of the freeway is the line equidistant from the two opposing faces, if they are parallel, or the bisector of the angle between the two faces, if they intersect. The boundary of the freeway is formed by the opposing faces of the two obstacles, along with extensions of these faces parallel to the spine. Figure 4 below shows a confined workspace with three polygonal obstacles and their associated freeways. The spines are shown as dotted lines.

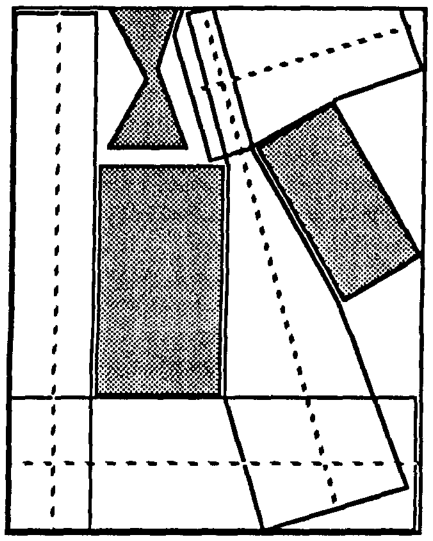

Figure 4. Freeways in a workspace with three obstacles.

The performance of the system was not acceptably fast even with the coarsest granularity of the grid. Computing the volume swept out by the robot took ten seconds. Once the free space tree had been determined, a trajectory could be calculated in three seconds. The example with the finest granularity had a calculation time of *eight hours* ! In spite of the disclaimer by the author that the purpose of the project was "to clarify and demonstrate the validity of the concepts", these execution times are more of a testament to the basic computational intractability of the complete determination of free space than they are a demonstration of the validity of the concepts.

Evaluation of the system's strengths and weaknesses. The strength of the paper is that it is a recent investigation into free space calculation. The author's system has the weaknesses that all free space calculation systems have: the assumption that obstacles are stationary and that the path of the manipulator, once determined, will not need to be re-computed for a considerable period of time.

Suitability of the system for use in a telerobotic environment. The system was not meant to be used in a telerobotic environment. Without major modification, it is not suitable.

## CONCLUSIONS

An exceptional amount of research has been done on the topic of collision avoidance, but the amount of work that has been done in the case of a telerobotic environment is small. Of the three general types of collision avoidance algorithms, the environment modelling and external control function systems hold the most promise for the telerobotic environment. Free space calculation algorithms, as a general class, are not suitable for the teleoperator environment, because they were designed for automatic planning of manipulator movements and the explicit calculation of free space poses a formidable computational burden. It would be interesting to determine if some free space calculation algorithm could be modified to calculate the free space only locally to decrease the computational burden and increase its flexibility.

The freeways are used as in the algorithm of Lozano-Pérez and Wesley to construct a collision-free path from a starting point to a goal point. Since the path is along the spines of the freeways, the path is centered between opposing faces of the obstacles.

Testing that the author performed on his system. The paper [7] describes only the implementation of the algorithm. No implementation or testing is mentioned. The description of his three-dimensional extension in [6] is at a high level and few concrete details are given. The illustrations in [6] imply that his algorithm was simulated in a three-dimensional environment. Brooks mentions that an example path was found in less than *one minute* on an MIT LISP machine.

Evaluation of the system's strengths and weaknesses. The strength of this system is its relative simplicity, although it is more complicated than the algorithm of Lozano-Pérez and Wesley. Brooks' algorithm has the advantage that the generated path stays as far as possible away from the obstacles. The weakness of the Brooks algorithm is its added complexity. Moreover, it shares the weakness of all of the free space calculation systems, in that the robot is assumed to be the only mobile object in the environment due to the high cost of re-calculating free space.

Suitability of the system for use in a telerobotic environment. All of the comments on the Lozano-Pérez and Wesley paper apply here also. No teleoperator is allowed for or needed in this algorithm. It is not meant to be used in a teleoperator environment.

In addition, the complexity of the algorithm contributes to an unacceptably high computational burden and slow update rate.

## Gouzènes

Description of the system. This paper [19] contains a discussion of the topological and geometric properties of free space and the configuration space of the robot, along with a method of calculating an approximation of free space. The main contribution of the paper is the idea of approximating the environment of the robot with a grid of rectangular parallelipipeds, and using this approximation to calculate both the configuration space of the robot and free space.

The basic method consists of calculating explicitly the exact volume swept out by the robot on its path, finding the center of this volume, growing the volume slightly to take care of inaccuracies in the approximation, and then checking for intersection of this volume with the obstacles.

Testing that the author performed on his system. The author implemented "a small two-dimensional geometric manipulation system" having three links. Part of the information about the volumes swept out by the links of the manipulator was entered by hand through a text editor. The system was tested for three different granularities of the approximation grid.

## APPENDIX
### Collision Avoidance Bibliography by Category

By its very nature, taxonomy is inexact. A certain amount of imprecision will always be involved in classifying objects into separate categories unless each object occupies a category of its own. The reader is offered apologies in advance for the inaccuracies that stem from separating the collision avoidance bibliography into only four different categories: *environmental modelling, external control function, free space calculation,* and *theoretical.* The first three of these categories are defined and discussed at length in the body of the paper. Only the fourth needs to be described here.

The *theoretical* category in this bibliography will be taken to include those papers that approach collision detection, collision avoidance, or path planning from an abstract, theoretical point of view. The theoretical analysis of the subject of the paper is the primary focus, with little or no interest in the practical implementation of the ideas involved. Solutions to the "piano movers" and the "ladder" problems are examples of this sort of paper. Some high-level path planning papers, concerned more with the aspects of *planning* the path than the path itself, also fit in this category.

The numbers of the items shown are those used in the bibliography.

### ENVIRONMENTAL MODELLING

2. Barker, L. Keith; Moore, Mary C., Theoretical Method for Calculating Relative Joint Geometry of Assembled Robot Arms, NASA Technical Paper 2155, NASA/Langley Research Center, 1983, 22 pp.

4. Boyse, John W. Interference Detection Among Solids & Obstacles. *Communications of ACM,* v. 22, no. 1 (Jan. 1979), pp. 3 – 9.

9. Cameron, S. A.; Culley, R. K. Determining the Minimum Translational Distance Between Two Convex Polyhedra, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 1, pp. 591 – 596.

12. Culley, R.K.; Kempf, K.G. A Collision Detection Algorithm Based on Velocity and Distance Bounds, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 2, pp. 1064 – 1069.

21. Herman, Martin. Fast, Three-Dimensional, Collision-Free Motion Planning, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 2, pp. 1056 – 1063.

38. Lumelsky, Vladimir. Continuous Motion Planning in Unknown Environment for a 3D Cartesian Robot Arm, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 2, pp. 1050 1055.

44. Oommen, B. John; Reichstein, Irvin. On Translating Ellipses Amidst Elliptic Obstacles, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 3, pp. 1755 – 1760.

45. Orlando, Nancy. An Algorithm for Collision Detection. Course Report, Computational Aspects of Robotics, Spring, 1985, 12 pp.

46. Orlando, Nancy. An Intelligent Robotics Control Scheme. *American Controls Conference,* San Diego, CA, June 6 - 8, 1984, 6 pp.

47. Orlando, Nancy. A System for Intelligent Teleoperation Research. *AIAA Computers in Aerospace IV Conference.* October 25, 1983, 6 pp.

52. Potter, Jerry. Robot Environment Expert System. NASA Contractor Report 3815, NAG-1-341 , NASA/Langley Research Center, 1985, 58 pp.

53. Requicha, Aristides A. G. Representations for Rigid Solids: Theory, Methods, and Systems, *ACM Computing Surveys,* v. 12, n. 4 (Dec. 1980), pp. 437 – 464.

55. Singh, J. Sanjiv; Wagh, Meghavad D. Robot Path Planning using Intersecting Convex Shapes: Analysis and Simulation. *IEEE J. Robotics and Automation,* v. RA–3, no. 2 (Apr. 1987), pp. 101 – 108.

57. Uchiki, Tetsuya; Ohashi, Toshiaki; Tokoro, Mario. Collision Detection in Motion Simulation. *Computers and Graphics,* v. 7, no. 3-4 (1983), pp. 285 – 293.

EXTERNAL CONTROL FUNCTION

3. Blubaugh, Thomas D. Time-Optimal Planning of Point-to-point Moves for Robotic Manipulators. Masters Thesis, Dept. of Mech. Engr. MIT, December, 1984, 257 pp.

8. Bruhr, H.; Ersü, E. Cartesian Path Planning by General Polynomial Interpolation. Conference Proceedings, 4 pp.

13. Dubowsky, S.; Norris, M. A.; Shiller, Z. Time Optimal Trajectory Planning for Robotic Manipulators with Obstacle Avoidance, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 3, pp. 1906 – 1912.

15. Freund, E.; Hoyer, H. Collision Avoidance in Multirobot Systems, *Robotics Research: The Second International Symposium,* (Aug. 20 – 23, 1984, Kyoto, Japan, Hanafus, I. & Inoue, H., eds.), MIT Press, 1985, pp. 135 – 146.

16. Freund, E.; Hoyer, H. On the On-line Solution to the Findpath Problem in Multi-Robot Systems, *Robotics Research: The Third International Symposium,* (Oct. 7 – 11, 1985, Gouvieux, France, Gaugeras, O. & Giralt, G. eds.), MIT Press, 1986, pp. 253 – 262

17. Freund, E.; Hoyer, H. Pathfinding in Multi-Robot Systems: Solution and Applications, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 1, pp. 103 – 111.

18. Freund, E.; Hoyer, H. Real-time Pathfinding in Multirobot Systems Including Obstacle Avoidance. *International J. Robotics Research,* v. 7, no. 1 (Feb. 1988), pp. 42 – 70.

23. Ish-Shalom, Jehuda. The CS Language Concept: A New Approach to Robot Motion Design. *IEEE Proceedings of 23rd Conference on Decision & Control,* (Las Vegas, NV, Dec. 1984). [also in *Int. J. Robotics Research,* v. 4, no. 1 (Spring 1985), pp. 760 – 767].

24. Khatib, O. Dynamic Control of Manipulators in Operational Space, *Proceedings of the Sixth World Congress on Theory of Machines and Mechanisms* (Dec. 15 – 20, 1983, New Delhi, India), pp 1128 – 1131.

25. Khatib, Oussama. The Operational Space Formulation in the Analysis, Design, and Control of Robot Manipulators, *Robotics Research: The Third International Symposium,* (Oct. 7 – 11, 1985, Gouvieux, France, Gaugeras, O. & Giralt, G. eds.), MIT Press, 1986, pp. 263 – 270.

26. Khatib, Oussama. Real-time Obstacle Avoidance for Manipulators & Mobile Robots. *International J. Robotics Research,* v. 5, no. 1 (Spring 1986), pp. 90 – 98.

27. Khatib, Oussama. A Unified Approach for Motion and Force Control of Robot Manipulators: The Operational Space Formulation. *IEEE J. Robotics and Automation,* v. RA–3, no. 1 (Feb. 1987), pp. 43 – 53.

28. Khatib, Oussama; Burdick, Joel. Motion and Force Control of Robot Manipulators, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 3, pp. 1381 – 1386.

30. Lee, B.H.; Lee, C.S.O. Collision-Free Motion Planning of Two Robots. *IEEE Trans. Systems, Man, Cybernetics,* v. SMC-17, no. 1 (Jan., Feb. 1987), pp. 21 – 32 (also in preprint form).

39. Lyons, Damian. Tagged Potential Fields: An Approach to Specification of Complex Manipulator Configurations, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 3, pp. 1749 – 1754.

42. Morris, Margaret Ann. Spatially Optimal Path Planning for Robotic Manipulators with Obstacle Avoidance and Joint Motion Constraints. B.S. Mech. Engr. Thesis at MIT (June, 1985), with Stephen Dubowsky.

50. Paul, Richard P.; Zhang, Hong. Robot Motion Trajectory Specification and Generation. ISRR Conference Proceedings, (Japan, 1984), pp. 373 – 379.

51. Pfieffer, Friedrich; Hohanni, Rainer. A Concept for Manipulator Trajectory Planning. *IEEE J. Robotics and Automation,* v. RA–3, no. 2 (Apr. 1987), pp. 115 – 123.

56. Suh, Suk-Hwan; Shin, Kang G. A Variational Dynamic Programming Approach to Robot-Path Planning with a Distance-Safety Criterion, *IEEE J. Robotics and Automation,* v. 4, no. 3 (June 1988), pp. 334 – 349.

FREE SPACE CALCULATION

5. Brooks, Rodney A. Find-Path for a PUMA-class robot. Conference proceedings, 5 pp.

6. Brooks, Rodney A. Planning Collision-Free Motions for Pick & Place Operations. MIT AI Lab, AI Memo #719, May 1983, 48 pp.

7. Brooks, Rodney A. Solving the Find-Path Problem by Good Representation of Free Space. *IEEE Trans. Systems, Man, Cybernetics,* v. SMC-13, no. 3 (Mar. – Apr. 1983), pp. 190 – 197.

10. Canny, John F. *The Complexity of Robot Motion Planning,* (1987 ACM Doctoral Dissertation Award), MIT Press, Cambridge, MA, 1988, 195 pp.

14. Erdmann, Michael; Lozano-Pérez, Tomás. On Multiple Moving Objects, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 3, pp. 1419 – 1424.

19. Gouzènes, Laurent. Strategies for Solving Collision-Free Trajectories Problems for Mobile and Manipulator Robots. *International J. Robotics Research,* v. 3, no. 4 (Winter 1987), pp. 51 – 65.

20. Hayward, Vincent. Fast Collision Detection Scheme by Recursive Decomposition of a Manipulator Workspace, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 2, pp. 1044 – 1049.

29. Krause, Donald M. Path Planning with Obstacle Avoidance. Part of grant report for NASA grant NAG-1-632, January 15, 1987 (with Dr. Mural Varanasi), 4 pp.

31. Lozano-Pérez, Tomás. Automatic Planning of Manipulator Transfer Movements. *IEEE Trans. Systems, Man, Cybernetics,* v. SMC-11, no. 10 (Oct. 1981), pp. 681 – 698.

32. Lozano-Pérez, Tomás. A Simple Motion-planning Algorithm for General Robotic Manipulators. *IEEE J. Robotics and Automation,* v. RA–3, no. 3 (June 1987), pp. 224 – 238.

33. Lozano-Pérez, Tomás. Spatial Planning: A Configuration Space Approach, *IEEE Trans. Computers,* v. C-32, no. 2 (Feb. 1983), pp. 108 – 120.

34. Lozano-Pérez, Tomás. Task Planning, chapter 6, pp. 473 – 493, of *Robot Motion: Planning and Control,* Michael Brady (ed.), MIT Press, 1982.

35. Lozano-Pérez, Tomás; Wesley, Michael. An Algorithm for Planning Collision-Free Paths among Polyhedral Obstacles. *Communications of ACM,* v. 22, no. 10 (Oct. 1979), pp. 560 – 570.

36. Luh, J.Y.S. A Scheme for Collision Avoidance with Minimum Distance Traveling for Industrial Robots. *J. Robotic Systems,* v. 1, no. 1 (1984), pp. 5 – 26.

37. Luh, J.Y.S.; Campbell, Charles E., Jr. Minimum Distance Collision-Free Path Planning for Industrial Robots with a Prismatic Joint. *IEEE Trans. Automatic Control,* v. AC-29, no. 8 (August, 1984), pp. 675 – 680.

58. Wallace, Richard Scot. A Digital Joint Space Approach to the Find-Path Problem for Manipulators. Preprint, 21 pp.

59. Wallace, Richard S. Three Find-path Problems. Conference proceedings, 4pp.

THEORETICAL

1. Baltsan, Avikam; Sharir, Micha. On the Shortest Paths between Two Convex Polyhedra. *Journal of the ACM,* v. 35, no. 2 (April 1988), pp. 267 – 287.

10. Canny, John F. *The Complexity of Robot Motion Planning,* (1987 ACM Doctoral Dissertation Award), MIT Press, Cambridge, MA, 1988, 195 pp.

11. Clarkson, Kenneth L. Approximation Algorithms for Shortest Path Motion Planning. *1987 Proceedings of the Symposium on the Theory of Computing,* pp. 56 – 65.

22. Hopcroft, J.E.; Joseph, D.A.; Whitesides, S.H. Movement Problems for Two-Dimensional Linkages. Tech. Report TR 82–515, Dept. of Computer Science, Cornell University, Ithaca, NY, 26 pp.

40. Mitchell, Joseph S.B. Shortest Rectilinear Paths among Obstacles. Tech. Report # 739 (Apr. 1987), School of O.R. & Indust. Engr., College of Engineering, Cornell University, Ithaca, NY, 45pp.

41. Mitchell, Joseph S.B.; Mount, David M.; Papadimitriou, Christos H. The Discrete Geodesic Problem. Preprint (Oct. 1985, rev. July 1986), Dept. of Operations Research, Stanford University, Palo Alto, CA, 28 pp.

43. Nagata, Tadashi; Honda, Kunihiko; Teramota, Yoshiaki. Multirobot Plan Generation in a Continuous Domain: Planning by Use of a Plan Graph and Avoiding Collisions among Robots. *IEEE J. Robotics and Automation*, v. RA–4, no. 1 (Feb. 1988), pp. 2 – 13.

48. Papadimitriou, Christos H.; Silverberg, Ellen B. Finding Feasible Paths for a Two-Point Body. Preprint (June 1987), 12 pp., Stanford University, Palo Alto, CA.

49. Papadimitriou, Christos H.; Silverberg, Ellen B. Optimal Piecewise Linear Motion of an Object among Obstacles. Preprint (June 1987), 17 pp., Dept. of O.R., Stanford University, Palo Alto, CA.

54. Roach, John W.; Boaz, Michael N. Coordinating the Motions of Robot Arms in a Common Workspace. *IEEE J. Robotics and Automation*, v. RA–3, no. 5 (Oct. 1987), pp. 437 – 444.

## BIBLIOGRAPHY

1. Baltsan, Avikam; Sharir, Micha. On the Shortest Paths between Two Convex Polyhedra. *Journal of the ACM*, v. 35, no. 2 (April 1988), pp. 267 – 287.

2. Barker, L. Keith; Moore, Mary C., Theoretical Method for Calculating Relative Joint Geometry of Assembled Robot Arms, NASA Technical Paper 2155, NASA/Langley Research Center, 1983, 22 pp.

3. Blubaugh, Thomas D. Time-Optimal Planning of Point-to-point Moves for Robotic Manipulators. Masters Thesis, Dept. of Mech. Engr. MIT, December, 1984, 257 pp.

4. Boyse, John W. Interference Detection Among Solids & Obstacles. *Communications of ACM*, v. 22, no. 1 (Jan. 1979), pp. 3 – 9.

5. Brooks, Rodney A. Find-Path for a PUMA-class robot. Conference proceedings, 5 pp.

6. Brooks, Rodney A. Planning Collision-Free Motions for Pick & Place Operations. MIT AI Lab, AI Memo #719, May 1983, 48 pp.

7. Brooks, Rodney A. Solving the Find-Path Problem by Good Representation of Free Space. *IEEE Trans. Systems, Man, Cybernetics*, v. SMC-13, no. 3 (Mar. – Apr. 1983), pp. 190 – 197.

8. Bruhr, H.; Ersü, E. Cartesian Path Planning by General Polynomial Interpolation. Conference Proceedings, 4 pp.

9. Cameron, S. A.; Culley, R. K. Determining the Minimum Translational Distance Between Two Convex Polyhedra, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 1, pp. 591 – 596.

10. Canny, John F. *The Complexity of Robot Motion Planning*, (1987 ACM Doctoral Dissertation Award), MIT Press, Cambridge, MA, 1988, 195 pp.

11. Clarkson, Kenneth L. Approximation Algorithms for Shortest Path Motion Planning. *1987 Proceedings of the Symposium on the Theory of Computing,* pp. 56 – 65.

12. Culley, R.K.; Kempf, K.G. A Collision Detection Algorithm Based on Velocity and Distance Bounds, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 2, pp. 1064 – 1069.

13. Dubowsky, S.; Norris, M. A.; Shiller, Z. Time Optimal Trajectory Planning for Robotic Manipulators with Obstacle Avoidance, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 3, pp. 1906 – 1912.

14. Erdmann, Michael; Lozano-Pérez, Tomás. On Multiple Moving Objects, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 3, pp. 1419 – 1424.

15. Freund, E.; Hoyer, H. Collision Avoidance in Multirobot Systems, *Robotics Research: The Second International Symposium,* (Aug. 20 – 23, 1984, Kyoto, Japan, Hanafus, I. & Inoue, H., eds.), MIT Press, 1985, pp. 135 – 146.

16. Freund, E.; Hoyer, H. On the On-line Solution to the Findpath Problem in Multi-Robot Systems, *Robotics Research: The Third International Symposium,* (Oct. 7 – 11, 1985, Gouvieux, France, Gaugeras, O. & Giralt, G. eds.), MIT Press, 1986, pp. 253 – 262

17. Freund, E.; Hoyer, H. Pathfinding in Multi-Robot Systems: Solution and Applications, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 1, pp. 103 – 111.

18. Freund, E.; Hoyer, H. Real-time Pathfinding in Multirobot Systems Including Obstacle Avoidance. *International J. Robotics Research,* v. 7, no. 1 (Feb. 1988), pp. 42 – 70.

19. Gouzènes, Laurent. Strategies for Solving Collision-Free Trajectories Problems for Mobile and Manipulator Robots. *International J. Robotics Research,* v. 3, no. 4 (Winter 1987), pp. 51 – 65.

20. Hayward, Vincent. Fast Collision Detection Scheme by Recursive Decomposition of a Manipulator Workspace, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 2, pp. 1044 – 1049.

21. Herman, Martin. Fast, Three-Dimensional, Collision-Free Motion Planning, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 2, pp. 1056 – 1063.

22. Hopcroft, J.E.; Joseph, D.A.; Whitesides, S.H. Movement Problems for Two-Dimensional Linkages. Tech. Report TR 82–515, Dept. of Computer Science, Cornell University, Ithaca, NY, 26 pp.

23. Ish-Shalom, Jehuda. The CS Language Concept: A New Approach to Robot Motion Design. *IEEE Proceedings of 23rd Conference on Decision & Control,* (Las Vegas, NV, Dec. 1984). [also in *Int. J. Robotics Research,* v. 4, no. 1 (Spring 1985), pp. 760 – 767].

24. Khatib, O. Dynamic Control of Manipulators in Operational Space, *Proceedings of the Sixth World Congress on Theory of Machines and Mechanisms* (Dec. 15 – 20, 1983, New Delhi, India), pp 1128 – 1131.

25. Khatib, Oussama. The Operational Space Formulation in the Analysis, Design, and Control of Robot Manipulators, *Robotics Research: The Third International Symposium,* (Oct. 7 – 11, 1985, Gouvieux, France, Gaugeras, O. & Giralt, G. eds.), MIT Press, 1986, pp. 263 – 270.

26. Khatib, Oussama. Real-time Obstacle Avoidance for Manipulators & Mobile Robots. *International J. Robotics Research,* v. 5, no. 1 (Spring 1986), pp. 90 - 98.

27. Khatib, Oussama. A Unified Approach for Motion and Force Control of Robot Manipulators: The Operational Space Formulation. *IEEE J. Robotics and Automation,* v. RA–3, no. 1 (Feb. 1987), pp. 43 – 53.

28. Khatib, Oussama; Burdick, Joel. Motion and Force Control of Robot Manipulators, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 - 10, 1986, San Francisco, California), v. 3, pp. 1381 – 1386.

29. Krause, Donald M. Path Planning with Obstacle Avoidance. Part of grant report for NASA grant NAG-1-632, January 15, 1987 (with Dr. Mural Varanasi), 4 pp.

30. Lee, B.H.; Lee, C.S.O. Collision-Free Motion Planning of Two Robots. *IEEE Trans. Systems, Man, Cybernetics,* v. SMC-17, no. 1 (Jan., Feb. 1987), pp. 21 - 32 (also in preprint form).

31. Lozano-Pérez, Tomás. Automatic Planning of Manipulator Transfer Movements. *IEEE Trans. Systems, Man, Cybernetics,* v. SMC-11, no. 10 (Oct. 1981), pp. 681 - 698.

32. Lozano-Pérez, Tomás. A Simple Motion-planning Algorithm for General Robotic Manipulators. *IEEE J. Robotics and Automation,* v. RA–3, no. 3 (June 1987), pp. 224 - 238.

33. Lozano-Pérez, Tomás. Spatial Planning: A Configuration Space Approach, *IEEE Trans. Computers,* v. C-32, no. 2 (Feb. 1983), pp. 108 – 120.

34. Lozano-Pérez, Tomás. Task Planning, chapter 6, pp. 473 - 493, of *Robot Motion: Planning and Control,* Michael Brady (ed.), MIT Press, 1982.

35. Lozano-Pérez, Tomás; Wesley, Michael. An Algorithm for Planning Collision-Free Paths among Polyhedral Obstacles. *Communications of ACM,* v. 22, no. 10 (Oct. 1979), pp. 560 – 570.

36. Luh, J.Y.S. A Scheme for Collision Avoidance with Minimum Distance Traveling for Industrial Robots. *J. Robotic Systems,* v. 1, no. 1 (1984), pp. 5 – 26.

37. Luh, J.Y.S.; Campbell, Charles E., Jr. Minimum Distance Collision-Free Path Planning for Industrial Robots with a Prismatic Joint. *IEEE Trans. Automatic Control*, v. AC-29, no. 8 (August, 1984), pp. 675 – 680.

38. Lumelsky, Vladimir. Continuous Motion Planning in Unknown Environment for a 3D Cartesian Robot Arm, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 2, pp. 1050 1055.

39. Lyons, Damian. Tagged Potential Fields: An Approach to Specification of Complex Manipulator Configurations, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 3, pp. 1749 – 1754.

40. Mitchell, Joseph S.B. Shortest Rectilinear Paths among Obstacles. Tech. Report # 739 (Apr. 1987), School of O.R. & Indust. Engr., College of Engineering, Cornell University, Ithaca, NY, 45pp.

41. Mitchell, Joseph S.B.; Mount, David M.; Papadimitriou, Christos H. The Discrete Geodesic Problem. Preprint (Oct. 1985, rev. July 1986), Dept. of Operations Research, Stanford University, Palo Alto, CA, 28 pp.

42. Morris, Margaret Ann. Spatially Optimal Path Planning for Robotic Manipulators with Obstacle Avoidance and Joint Motion Constraints. B.S. Mech. Engr. Thesis at MIT (June, 1985), with Stephen Dubowsky.

43. Nagata, Tadashi; Honda, Kunihiko; Teramota, Yoshiaki. Multirobot Plan Generation in a Continuous Domain: Planning by Use of a Plan Graph and Avoiding Collisions among Robots. *IEEE J. Robotics and Automation*, v. RA–4, no. 1 (Feb. 1988), pp. 2 – 13.

44. Oommen, B. John; Reichstein, Irvin. On Translating Ellipses Amidst Elliptic Obstacles, *Proceedings of 1986 IEEE International Conf. on Robotics & Automation* (Apr. 7 – 10, 1986, San Francisco, California), v. 3, pp. 1755 – 1760.

45. Orlando, Nancy. An Algorithm for Collision Detection. Course Report, Computational Aspects of Robotics, Spring, 1985, 12 pp.

46. Orlando, Nancy. An Intelligent Robotics Control Scheme. *American Controls Conference*, San Diego, CA, June 6 - 8, 1984, 6 pp.

47. Orlando, Nancy. A System for Intelligent Teleoperation Research. *AIAA Computers in Aerospace IV Conference.* October 25, 1983, 6 pp.

48. Papadimitriou, Christos H.; Silverberg, Ellen B. Finding Feasible Paths for a Two-Point Body. Preprint (June 1987), 12 pp., Stanford University, Palo Alto, CA.

49. Papadimitriou, Christos H.; Silverberg, Ellen B. Optimal Piecewise Linear Motion of an Object among Obstacles. Preprint (June 1987), 17 pp., Dept. of O.R., Stanford University, Palo Alto, CA.

50. Paul, Richard P.; Zhang, Hong. Robot Motion Trajectory Specification and Generation. ISRR Conference Proceedings, (Japan, 1984), pp. 373 – 379.

51. Pfieffer, Friedrich; Hohanni, Rainer. A Concept for Manipulator Trajectory Planning. *IEEE J. Robotics and Automation,* v. RA–3, no. 2 (Apr. 1987), pp. 115 – 123.

52. Potter, Jerry. Robot Environment Expert System. NASA Contractor Report 3815, NAG-1-341 , NASA/Langley Research Center, 1985, 58 pp.

53. Requicha, Aristides A. G. Representations for Rigid Solids: Theory, Methods, and Systems, *ACM Computing Surveys,* v. 12, n. 4 (Dec. 1980), pp. 437 – 464.

54. Roach, John W.; Boaz, Michael N. Coordinating the Motions of Robot Arms in a Common Workspace. *IEEE J. Robotics and Automation,* v. RA–3, no. 5 (Oct. 1987), pp. 437 – 444.

55. Singh, J. Sanjiv; Wagh, Meghavad D. Robot Path Planning using Intersecting Convex Shapes: Analysis and Simulation. *IEEE J. Robotics and Automation,* v. RA–3, no. 2 (Apr. 1987), pp. 101 – 108.

56. Suh, Suk-Hwan; Shin, Kang G. A Variational Dynamic Programming Approach to Robot-Path Planning with a Distance-Safety Criterion, *IEEE J. Robotics and Automation,* v. 4, no. 3 (June 1988), pp. 334 – 349.

57. Uchiki, Tetsuya; Ohashi, Toshiaki; Tokoro, Mario. Collision Detection in Motion Simulation. *Computers and Graphics,* v. 7, no. 3-4 (1983), pp. 285 – 293.

58. Wallace, Richard Scot. A Digital Joint Space Approach to the Find-Path Problem for Manipulators. Preprint, 21 pp.

59. Wallace, Richard S. Three Find-path Problems. Conference proceedings, 4pp.